Unstructured Mesh PIC on Accelerated Systems

Cameron W. Smith, Gerrett Diamond,
Gopan Perumpilly, Onkar Sahni, Mark S. Shephard

Rensselaer Polytechnic Institute

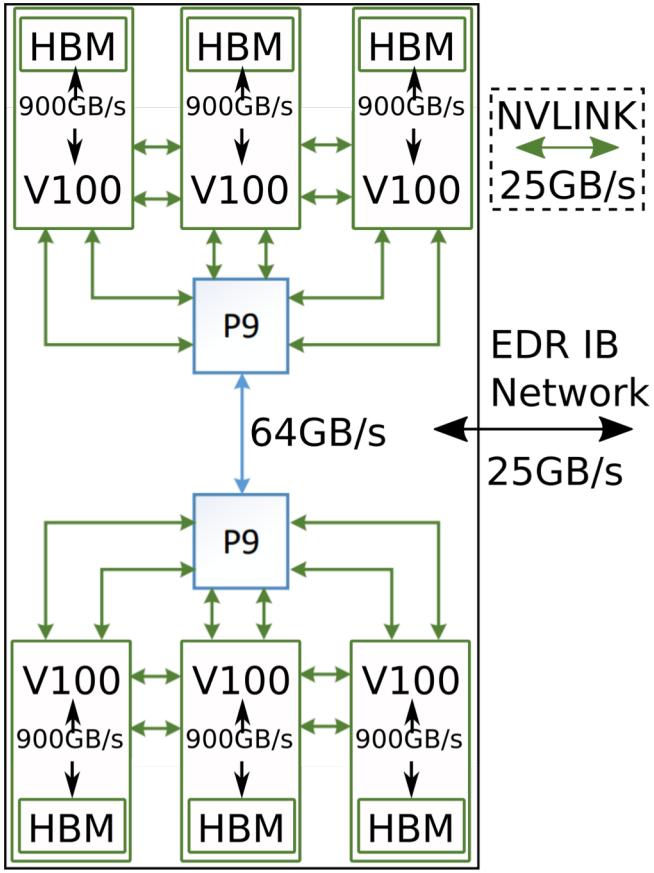
FASTMath All-hands Meeting
June, 11th 2019

Summit Computing Hardware

Focus on GPUs - they provide 95% of memory bandwidth and 98% of FLOPs

- 6 V100 and 2 P9 per node
- 4,608 nodes → 27,648 V100 and 9,216 P9

Processor	Double Precision TeraFLOPs	Memory Bandwidth (GB/s)
V100 vs P9		
V100	7.5	900
P9	0.5	135
P9/V100	7%	15%
Full System		
V100	207,360	24,833,200
P9	4,608	1,244,160
P9/V100	2%	5%

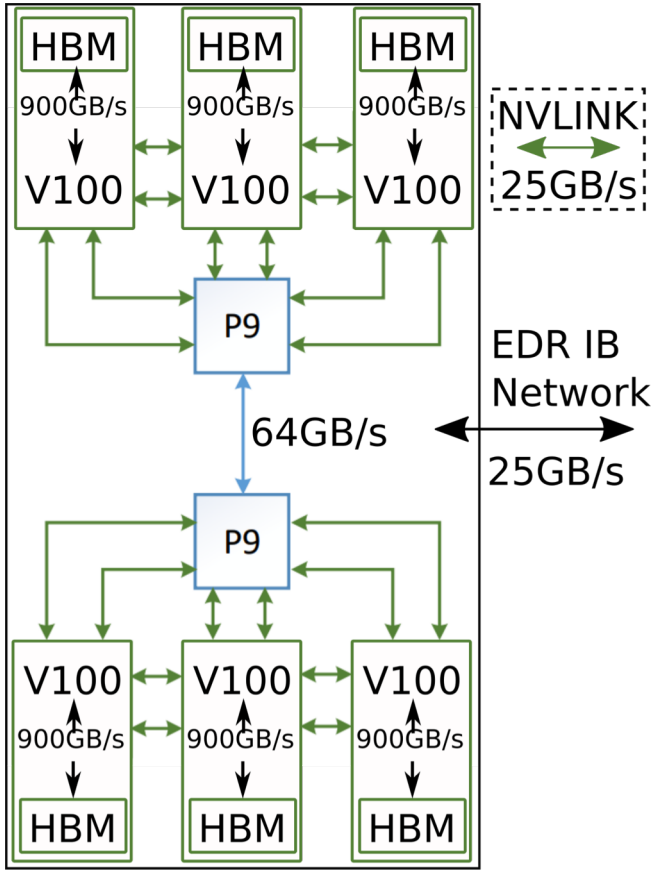


J.Choquette. Hot Chips 2017. "Volta: Programmability and Performance"

Summit Communication Hardware

Overlap communication with computation and minimize inter-node communication:
205:1 ratio of aggregate V100 HBM bandwidth to inter-node EDR bandwidth

System	Stream Triad (GB/s)	Network Peak (GB/s)	Stream/ Network
Summit (inter-node)	5,130	25	205
Summit (intra-node)	855	50	17
Stampede2	194	12	17
Mira	27	20	1.4
Perlmutter (inter-node)[1]	5,130	100	51



Summit stream/network ratios

- inter-node - sum six V100 HBM to EDR IB bandwidth
- intra-node - V100 HBM bandwidth to NVLINK bandwidth

[1] '4x Volta Next' ?= 4x1.5xV100, Slingshot 4x 25GB/s links

Load Balancing on Accelerated Systems

Summit GPUs provide > 90% of the system performance

- Technically heterogeneous...
- Development cost to use CPU thread/vector parallelism in addition to GPU parallelism is not worth the performance benefit

Focus on providing a `decent' load imbalance to GPUs with minimized induced inter-GPU data movement

- Placing processes near those they share domain data with
- Optimizing partitions for communication, possibly sacrificing load balance
- Predictively load balance to reduce calls to balancer

Many more challenges if we had a processor/node with multiple specialized accelerators (e.g., compression, fft, spmv, graph traversal, encryption, etc...):

- Will current programming models/tools work effectively? Major code rewrite?
- Many post exascale technologies in the pipeline: www.crnch.gatech.edu/sites/default/files/02-siamcse-2019-shalf.pdf

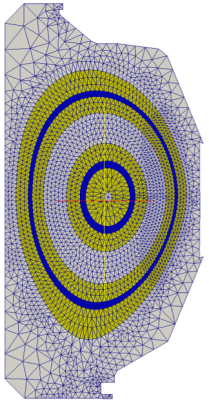
Basics of a Mesh-Based PIC Approach

Mesh distribution – PICParts

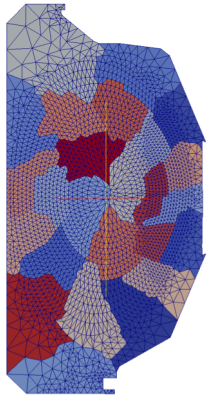
- A part from a partitioned mesh is the core part
- Additional parts surrounding core part satisfy data dependencies
- Particles can be migrated – satisfy dependencies, maintain load balance
- Using Omega from Sandia National Labs – GPU ready, tested on AC922 (Summit/Sierra)

Particle data structure

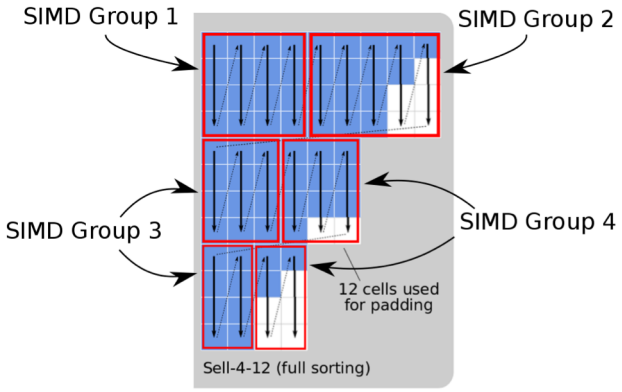
- Groups particles by mesh element
- Optimizes push, scatter, gather
- Rotated and sorted CSR; a row has the particles of an element
- Working with COPA Cabana team



Two XGC PICparts



Mesh Partition



SCS with vertical slicing
Besta, Marending, Hoefler, IPDPS 2017

Pseudo Push Test: Large Particle Counts on GPU

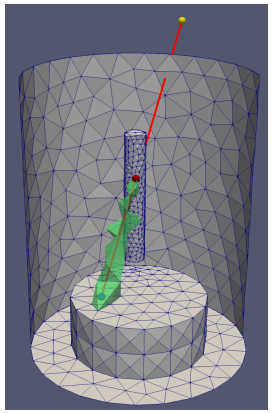
Executed 20 push, search, rebuild iterations in pisces mesh (~6k tets)

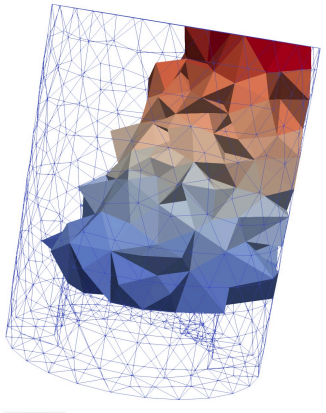
- Particles initialized off of bottom inset face
- One process and one GPU on RPI DCS System – node P9 (x2) and GV100 (x4)

Running larger particle counts and multiple processes – testing:

- Parallel PICPart creation
- Bulk communication layer w/MPI wrapper for MPI, CUDA aware MPI, and possibly NCCL
- Integrating GITRm physics - Boris move, particle-boundary interactions, near-boundary fields, field-following meshes

	no sorting	full sorting
ptcls (Ki)	time (s)	time (s)
128	2.298661	3.642041
256	2.895464	3.415048
512	3.79263	3.851178
1024	4.972283	4.090044
2048	7.089673	4.389198
4096	11.578984	4.799475





(left) Path of a particle through mesh. (right) Elements colored by iteration when particles enter; 0=blue, 20=red.